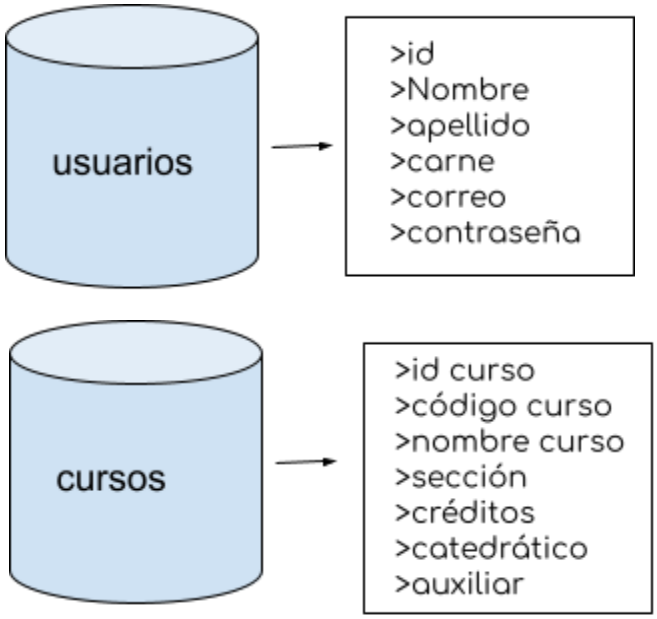
USAC
TRICENTENARIA
Universidad de San Carlos de Guatemala

Manual técnico

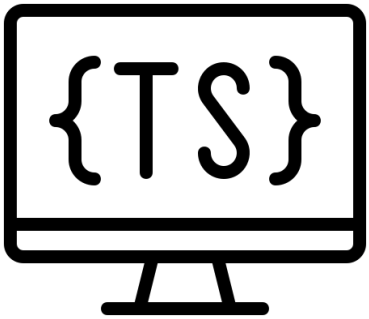
Grupo #13
Austin Antonio Álvarez Medina - 201906143
Oswaldo Antonio Choc Cúteres - 201901844
Juan José Gerardi Hernandez - 201900532

El programa utiliza diversas tecnologías para lo que es el backend,frontend y la base de datos.

- Base de datos: Para las bases de datos se utilizó Mysql el cual sirve para almacenar los datos de los usuarios y de los cursos.



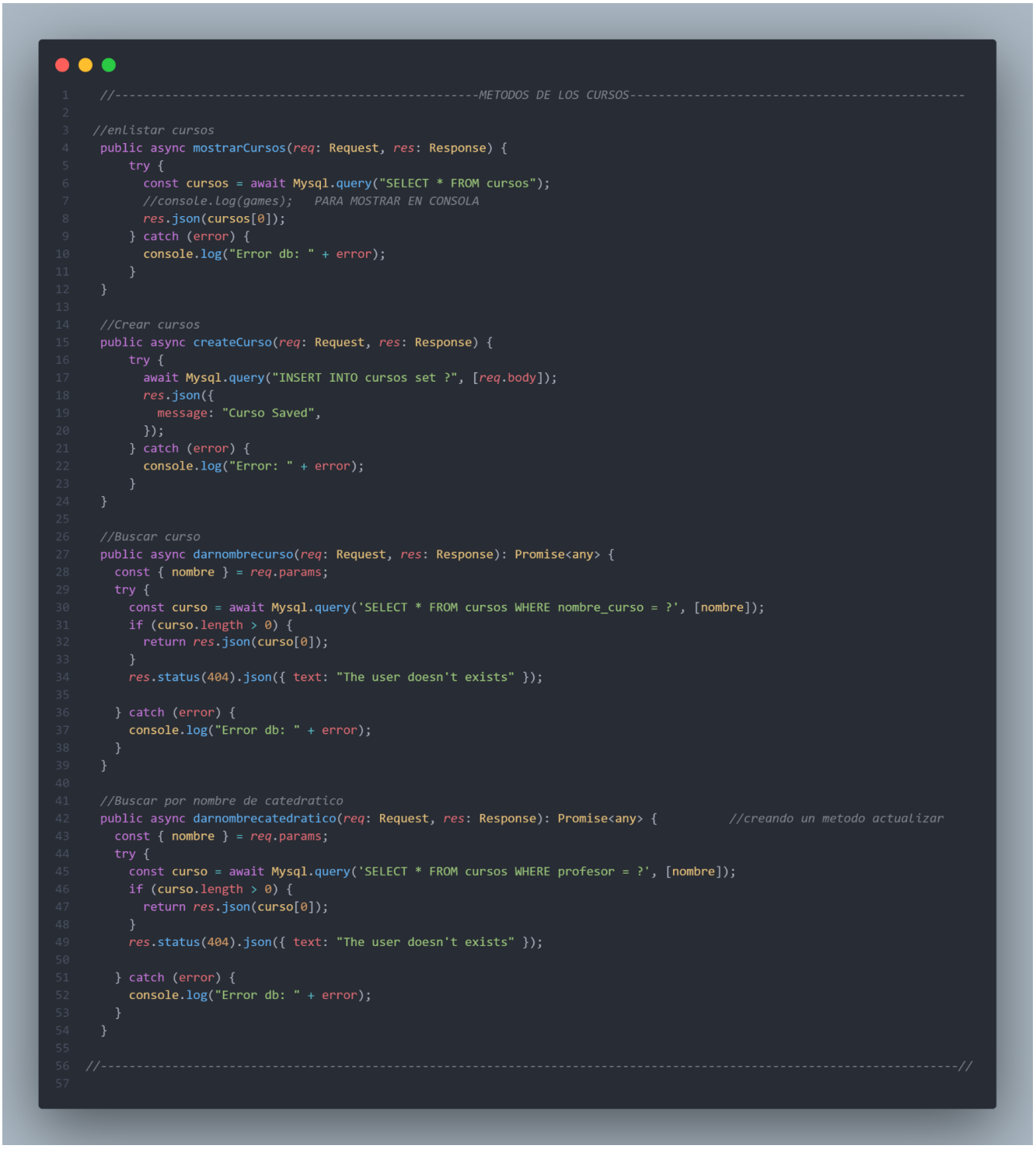
- Backend: En el lado del backend se utilizo lo que es nodejs y typescript para realizar el api rest donde se realizan las peticiones get, put, post, delete para ver, editar, crear y borrar respectivamente los datos de las bases de datos.



> endpoints para realizar consulta, editar password y guardar usuarios.

```
//------------------------------------------------ METODOS DE LOS USUARIOS-----------------------------------------
  //enlistar usuarios
  public async list(req: Request, res: Response) {
      try {
        const users = await Mysql.query("SELECT * FROM users");
          //console.log(games);   PARA MOSTRAR EN CONSOLA
          res.json(users[0]);
      } catch (error) {
          console.log("Error db: " + error);
      }
  }
  //Guardar usuarios
  public async createUsuario(req: Request, res: Response) {
      try {
        await Mysql.query("INSERT INTO users set ?", [req.body]);
        res.json({
          message: "User Saved",
        });
      } catch (error) {
        console.log("Error: " + error);
      }
  }
  //cambiar password
  public async singin(req: Request, res: Response){
    const {carne, contrasena} = req.body;
      try {
        const valores = await Mysql.query('SELECT * from users WHERE carne=? and contrasena=?',[carne,contrasena],)

        if(valores.length > 0){
          return res.json(valores[0]);

        }

        return res.status(404).json({text: "The user doesn't exists"});

    } catch (error) {
      console.log("Error db: " + error);
    }
  }

```

> endpoints para enlistar y crear cursos, buscar cursos por nombre, buscar cursos por nombre de catedrático.



```
//----------------------------------------------------METODOS DE LOS CURSOS-----------------------------------------------

//enlistar cursos
public async mostrarCursos(req: Request, res: Response) {
    try {
      const cursos = await Mysql.query("SELECT * FROM cursos");
      //console.log(games);   PARA MOSTRAR EN CONSOLA
      res.json(cursos[0]);
    } catch (error) {
      console.log("Error db: " + error);
    }
}

//Crear cursos
public async createCurso(req: Request, res: Response) {
    try {
      await Mysql.query("INSERT INTO cursos set ?", [req.body]);
      res.json({
        message: "Curso Saved",
      });
    } catch (error) {
      console.log("Error: " + error);
    }
}

//Buscar curso
public async darnombrecurso(req: Request, res: Response): Promise<any> {
  const { nombre } = req.params;
  try {
    const curso = await Mysql.query('SELECT * FROM cursos WHERE nombre_curso = ?', [nombre]);
    if (curso.length > 0) {
      return res.json(curso[0]);
    }
    res.status(404).json({ text: "The user doesn't exists" });

  } catch (error) {
    console.log("Error db: " + error);
  }
}

//Buscar por nombre de catedratico
public async darnombrecatedratico(req: Request, res: Response): Promise<any> {          //creando un metodo actualizar
  const { nombre } = req.params;
  try {
    const curso = await Mysql.query('SELECT * FROM cursos WHERE profesor = ?', [nombre]);
    if (curso.length > 0) {
      return res.json(curso[0]);
    }
    res.status(404).json({ text: "The user doesn't exists" });

  } catch (error) {
    console.log("Error db: " + error);
  }
}

//-------------------------------------------------------------------------------------------------------------------------//

```

> rutas a las cuales se mandan las solicitudes de cursos y usuarios.

```ts
import { Router } from 'express';
import usersController from '../controllers/UserController';

class UsersRoutes {
    public router: Router = Router();

    constructor(){
        this.config();
    }

    config(): void{

        // URL DE LOS METODOS DE USUARIOS
        this.router.get('/',usersController.list );
        this.router.post('/singin', usersController.singin);
        this.router.post('/', usersController.createUsuario);


        // URL DE LOS METODOS PARA CURSOS
        this.router.get('/cursos',usersController.mostrarCursos );
        this.router.post('/cursos',usersController.createCurso);
        this.router.delete('/cursos/:id',usersController.delete);
        this.router.get('/cursos/cate/:nombre',usersController.darnombrecatedratico);
        this.router.get('/cursos/:nombre',usersController.darnombrecurso);
        this.router.put('/:id',usersController.update); //NO FUNCIONA
    }

}

const usersRoutes = new UsersRoutes();
export default usersRoutes.router;
```

- Frontend: en la parte de fronten se utilizaron tecnologías como Angular el cual es un framework para crear aplicaciones web el cual utiliza typescript, bootstrap el cual es una biblioteca para diseños de estilos, html y css.

Código html para crear el login de la aplicación con estilos obtenidos de bootstrap.

```
<app-navigation></app-navigation>
<br>

    <div class="col-md-6 offset-md-3">          <!--DISEÑO DE LA PESTAÑA ADD AGREGANDO LOS TEXTAREA-->
    <div class="card">
        <div class="card-body ">
            <form >

                <img class="mx-auto d-block" src="/assets/Logo.JPG" alt="Logo" width="350">

                <br>
                <div class="form-control">
                    <input type="text" name = "carnet" placeholder="CUI/REGISTRO ACADEMICO" [(ngModel)] = "user.carne"  class="form-control" required>
                </div>

                <br>
                <div class="form-control">
                    <input type="password" name = "password" placeholder="Password" [(ngModel)] = "user.contrasena" class="form-control" required>
                </div>

                <br>
                    <button type="submit" class="btn btn-success w-100" (click) = "login()">Login</button>

                <br><br>


                <div class="col-auto text-center">
                <a class="navbar-brand " routerLink ="/Login/creat"><i class="fas fa-user-plus"></i>  Crear Cuenta   </a>
                <a class="navbar-brand " routerLink ="/Login/recuperar">    ¿Olvidaste la contraseña?</a>
                </div>
            </form>
        </div>
    </div>
</div>

```

Rutas de las pestañas de la aplicación “app-routing.modules.ts”.

```ts
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import {UserFormComponent} from './components/user-form/user-form.component';
import {LoginComponent} from './components/login/login.component';
import { RecuperarCuentaComponent } from './components/recuperar-cuenta/recuperar-cuenta.component';
import { PaginaPrincipalComponent } from './components/pagina-principal/pagina-principal.component';
import { PublicacionesComponent } from './components/publicaciones/publicaciones.component'
import { PerfilComponent } from './components/perfil/perfil.component'

//DONDE SE CREAN LAS RUTAS
const routes: Routes = [
  {
    path: ' ',
    redirectTo: '/Login',
    pathMatch: 'full'
  },

  //------------------------------------------- URL USUARIOS ----------------------------------------------
  {
    path: 'Login',
    component: LoginComponent
  },

  {
    path: 'Login/creat',
    component: UserFormComponent
  },

  {
    path: 'Login/recuperar',
    component: RecuperarCuentaComponent
  },

// ------------------------------------------------- URL CURSOS-------------------------------------------------

{
  path: 'pageP',
  redirectTo:'/pageP',
  pathMatch: 'full'
},
{
    path: 'pageP',
    component: PaginaPrincipalComponent,
},
{
  path:'pageP/publi',
  component: PublicacionesComponent
},
{
  path: 'pageP/miP',
  component: PerfilComponent
}


];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }

```